TO: USPTO

ZILKA·KOTAB PC

ZILKA, KOTAB & FEECE™

RECEIVED
CENTRAL FAX CENTER

AUG 2 3 2006

100 PARK CENTER PLAZA, SUITE 300
SAN JOSE, CA 95113

TELEPHONE (408) 971-2573
FAX (408) 971-4660

FAX COVER SHEET

Date:	August 23, 2006	**Phone Number**	**Fax Number**
To:	Examiner Homayounmehr		(571) 273-8300
From:	Kevin J. Zilka		

Docket No.: NAI1P317/01.185.01 **Application No.: 10/091,645**

Total Number of Pages Being Transmitted, Including Cover Sheet: 19

Message:

Please deliver to Examiner Homayounmehr.

Thank you,

Kevin J. Zilka

☐ *Original to follow Via Regular Mail* X *Original will Not be Sent* ☐ *Original will follow Via Overnight Courier*

IF YOU DO NOT RECEIVE ALL PAGES OR IF YOU ENCOUNTER
ANY OTHER DIFFICULTY, PLEASE PHONE Erica
AT (408) 971-2573 AT YOUR EARLIEST CONVENIENCE

August 23, 2006

RECEIVED
CENTRAL FAX CENTER
AUG 2 3 2006

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	)
Handong Wu et al.	) Group Art Unit: 2132
Application No. 10/091,645	) Examiner: Homayounmehr, Farid
Filed: 03/05/2002	) Date: 08/23/2006
For: NETWORK INTRUSION DETECTION AND ANALYSIS SYSTEM AND METHOD	)

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

ATTENTION: Board of Patent Appeals and Interferences

REPLY BRIEF (37 C.F.R. § 41.37)

This Reply Brief is being filed within two (2) months of the mailing of the Examiner's Answer mailed on 06/23/2006.

Following is an issue-by-issue reply to the Examiner's Answer.

CERTIFICATE OF MAILING/TRANSMISSION (37 C.F.R. § 1.8(a))

I hereby certify that this correspondence is, on the date shown below, being:

MAILING

☐ deposited with the United States Postal Service with sufficient postage as first class mail, in an envelope addressed to the Commissioner for Patents, Washington, D.C. 20231.

Date: 8/23/2006

FACSIMILE

☒ transmitted by facsimile to the Patent and Trademark Office, (571) 273-8300.

Signature

Erica L. Farlow

(type or print name of person certifying)

Issue # 1:

The Examiner has rejected Claims 21 and 22 under 35 U.S.C. 112, first paragraph, as not being enabled. With respect to Claim 21, the Examiner has questioned the exact implication of the "frame_context_pointer_position" limitations. With respect to Claim 22, the Examiner has stated that Page 12 of the specification only mentions the incision of "frame_tcp_bridge," "frame_udp_bridge," "frame_ip_bridge," and "frame_http_bridge," but does not give a description of the specific functionality of such elements.

Appellant respectfully asserts that Page 12 of appellant's disclosure describes the form the API's may take, or, in other words, defines the form that the API takes. Thus, according to the claims, the API is defined according to "frame_context_pointer_position" (Claim 21) which includes "frame_tcp_bridge; frame_udp_bridge; frame_ip_bridge; and frame_http_bridge" (Claim 22).

In the Examiner's Answer mailed 06/23/2006, the Examiner argued that "[d]efining a form of an API, as it appears in Page 12 of the specification, is not enabling, as it does not disclose what functionality is being communicated on requests between the intrusion detection device and the data monitoring device." Appellant respectfully disagrees and asserts that Page 12 of appellant's disclosure describes that "[t]he APIs 48 are used to parse, generate, and load signatures, invoke corresponding signature detection methods from appropriate protocol contexts, access states required for stateful intrusion detection, and access alerts/alarms management facilities." In addition, appellant respectfully asserts that item 48 on Figure 1 and Figure 3, and the accompanying description, would enable one skilled in the art to make and use such an API that is claimed in Claims 21 and 22.

It should be strongly noted that the foregoing citations do not in any way state, suggest, or otherwise imply that such claim language is limited in scope to the specific embodiments depicted in the drawings or described in the specification in the instant application.

Issue # 2:

The Examiner has rejected Claims 21 and 22 under 35 U.S.C. 112, second paragraph, as failing to define the invention. Appellant respectfully disagrees. In the Examiner's Answer mailed 06/23/2006, the Examiner argued that "[t]he functionality of API's such as frame_context_pointer_position, frame_tcp_bridge, frame_udp_bridge, frame_ip_bridge, [and] frame_http_bridge is not defined anywhere in the disclosure" and that the "[c]laims are indefinite based in the lack of enablement expressed above." Appellant respectfully disagrees and references the arguments made hereinabove with respect to enablement.

Issue # 3:

The Examiner has rejected Claims 1-20 under 35 U.S.C. 102(e) as being anticipated by or, in the alternative, unpatentable under 35 U.S.C. 103(a) as being obvious over Vaidya (U.S. Patent No. 6,279,113) in view of Porras (U.S. Patent Application No. 2003/0101358).

Group #1: Claims 1-8, 10, 14-15, and 18-20

Claim Element #1

With respect to each of the independent claims, and specifically appellant's claimed "intrusion detection device separate from the data monitoring device," the Examiner has argued, in the Office Action dated 11/30/2005, that appellant's claimed "intrusion detection device separate from the data monitoring device" is only separate in functionality. Appellant respectfully points out page 7, line 14-page 8, line 6, along with associated Figure 1, which clearly shows that the network analysis and data monitoring device 16 and the intrusion detection device 14 are separate devices, and not merely that they perform separate functionality, as the Examiner contends.

In the Examiner's Answer mailed 06/23/2006, the Examiner argued "[a]ppellant's element 18 is shown in Figure 2 as one item called NIDAS in the network" and "[t]herefore, the separation described in disclosure implies nothing more than a functional separation, and the broad interpretation of claims and figures is that the system is within on element (NIDAS, device 18 of Figure 2)." The Examiner further argued that "functional separation is not excluded in the claim language." Appellant respectfully asserts that item 18 of Figure 1 is referred to, on Page 7 of appellant's disclosure, as "a network intrusion detection system" which "provides an intrusion detection device 14 in combination with a network analysis and data monitoring device 16" (emphasis added). Clearly, appellant's disclosure that the NIDAS system includes a separate intrusion detection device and a separate network analysis and data monitoring device suggests that the separation is not just functional as argued by the Examiner.

In addition, appellant respectfully disagrees with the Examiner's assertion that "Vaidya discloses [in] the claims... a system including functionally separate devices" since Col. 6, lines 1-11 disclose that "[t]he configuration builder module 32 accesses the appropriate attack signature profile sets during operation of the data collector 10 and provides the attack signature profiles to a stateful dynamic signature inspection (SDSI) virtual processor 36" (emphasis added). Clearly, the mere disclosure that the data collector 10 provides attack signatures profiles to a SDSI virtual processor 36, which is included within data collector 10, fails to even suggest an "intrusion detection device [which is] separate from the data monitoring device" (emphasis added), as claimed by appellant.

The Examiner has also argued that Vaidya does not limit his invention to one processor only. In making such an assertion, the Examiner has referenced Figure 4, items 36, 34 and 38 as being separate modules to perform separate functionalities. Appellant respectfully asserts that Figure 4 only discloses modules that work with the virtual processor, but not that such modules are separate processors. Thus, appellant respectfully asserts that the only processing device in Vaidya is the virtual processor. Furthermore, the modules relied on by the Examiner do not provide the

separate functionality claimed by appellant, namely "captur[ing] data passing through the network," "perform[ing] intrusion detection," etc.

In the Examiner's Answer mailed 06/23/2006, the Examiner argued that 'the general meaning of the word "processors," which in the context of the subject at hand, is an element capable of processing data and performing operations.' Appellant respectfully disagrees with the Examiner's argument and asserts that Vaidya merely discloses that "[a]lthough a preferred embodiment of the processor employs the software based virtual processor 36 to execute attack signature profiles, a hardware based processor can be employed in the place of the virtual processor 36" (emphasis added). Clearly, Vaidya discloses a processor as a software based virtual processor or a hardware based processor.

The Examiner's argues that "the purpose for referencing items 36, 34 and 38 in the second office action was to indicate that Vaidya is not limited to one processor, and discloses use of separate processors performing different parts of operation." However, appellant respectfully asserts that item 10 of Figure 2 is disclosed as a "data collector 10 [which] includes a communication module 34," "a configuration builder module 32," "a stateful dynamic signature inspection (SDSI) virtual processor 36," and a "reaction module 38" (Col. 6, lines 1-26). Clearly, the mere disclosure that data collector 10 includes a virtual processor and several other modules simply fails to even suggest an "intrusion detection device separate from the data monitoring device" (emphasis added), as claimed by appellant.

Still yet, the Examiner has argued that Vaidya performs the functionality of appellant's data monitoring device and intrusion detection device in item 36, but that such functionality is separate as shown in item 40. Appellant respectfully asserts that item 40, the register cache, "temporarily stores information extracted from a data packet which determines which signature profile(s) will be accessed from the signature profile memory 39." Clearly, such register cache that only stores information from data packets does not meet appellant's claimed "data monitoring device," which specifically "capture[s] data passing through the network," "monitor[s] network traffic," "decode[s] protocols for grouping packets into

different protocol presentations and assembling the packets into high level protocol groups," and "analyze[s] received data," in the manner claimed by appellant. Thus, the functionality of items 36 and 40, as relied on by the Examiner, does not meet appellant's specific claim language.

In the Examiner's Answer mailed 06/23/2006, the Examiner argued that "[r]eferring to figure 4 again, Vaidya performs the functionality of applicant's data monitoring device and intrusion detection device in item 36, however, within item 36 it discloses item 40 as a separate device." In addition, the Examiner argues that "Vaidya clearly separates the functionality of data collection, as described in Fig 5, from intrusion detection, as described in Figures 6 to 10."

Appellant respectfully disagrees with the Examiner's arguments that Vaidya discloses separate data collection and intrusion detection. Appellant respectfully asserts that Vaidya merely discloses that "[w]ith reference to FIG. 4, the operation of the virtual processor 36 includes monitoring network data 46 to determine whether the data is associated with a network intrusion" and that "[a] register cache 40 temporarily stores information extracted from a data packet which determines which signature profile(s) will be accessed from the signature profile memory 39" (Col. 7, lines 12-18 – emphasis added). In addition, Vaidya discloses that "[t]he virtual processor 36 obtains a data packet from a queue and extracts MAC header information, IP header information, transport header information, and application information from the data packet" (Col. 7, lines 18-21 – emphasis added). Further, Vaidya discloses that "[r]eferring to FIG. 5, a method [is shown] for building a register cache 40 during the operation of the virtual processor 36" and that "[a]ll of the extracted packet information is entered into the register cache 40" (Col. 8, lines 40-49 – emphasis added). Also, Vaidya discloses that "[t]he virtual processor 36 processes the attack signature profiles to determine whether the packet is associated with a network intrusion attempt" (Col. 7, lines 32-36 – emphasis added).

However, the mere disclosure by Vaidya that the virtual processor 36 monitors network data 46 and stores extracted data packet information into the register cache 40, and that the virtual processor 36 determines if the packet is associated with a network intrusion

attempt simply fails to even suggest an "intrusion detection device separate from the data monitoring device" (emphasis added), as claimed by appellant. Clearly, the virtual processor 36 in Vaidya is performing both the data monitoring, information extraction from data packets, and intrusion detection which simply fails to even suggest an "intrusion detection device separate from the data monitoring device" (emphasis added), as claimed by appellant.

Furthermore, the Examiner has argued that Vaidya's claim 1, which claims a method of detecting intrusion attempts, is broken down into several steps, including monitoring network traffic and network intrusion. Appellant emphasizes that merely claiming separate steps does not meet appellant's separate devices, as claimed. In addition, simply claiming monitoring network traffic, as in Vaidya, does not meet appellant's specifically claimed functionality of a data monitoring device that exceeds beyond monitoring network traffic, as excerpted above.

In the Examiner's Answer mailed 06/23/2006, the Examiner argued that "the purpose of Examiner's assertion that Vaidya performs operations in separate steps was to establish that Vaidya inherently discloses separate modules performing separate functionalities." In addition, it was argued that the "Examiner has not merely relied on separate steps to meet appellant separate devices." Appellant respectfully asserts that item 10 of Figure 2 in Vaidya is disclosed as a "data collector 10 [which] includes a communication module 34," "a configuration builder module 32," "a stateful dynamic signature inspection (SDSI) virtual processor 36," and a "reaction module 38" (Col. 6, lines 1-26). Clearly, the mere disclosure that data collector 10 includes a virtual processor and several other modules simply fails to even suggest an "intrusion detection device separate from the data monitoring device" (emphasis added), as claimed by appellant.

Appellant again respectfully asserts that appellant's arguments made in the Amendment A dated 10/12/2005 on page 7, paragraph 4-page 9, paragraph 1 clearly show the distinction between Vaidya and appellant's specific claim language. In addition, the Examiner's response in the Office Action dated 11/30/2005 failed to provide a prior art showing of appellant's claimed technique for the reasons argued hereinabove.

Claim Element #2

With respect to appellant's claimed technique "wherein the application program interfaces allow the intrusion detection device to leverage the separate data monitoring device, by allowing the intrusion detection device to call an application program interface configured to open a protocol decoding application associated with the separate data monitoring device, and by allowing the intrusion detection device to call an application program interface configured to open an alarm generation application associated with the separate data monitoring device," the Examiner has argued that such claim language in addition to the specification does not point out any advantage in separation of the intrusion detection device and the data monitoring device. In response, appellant points out page 8, lines 3-6 which states that the components, including the intrusion detection device and the network analysis and data monitoring device can perform dual simultaneous functions, etc. which allows for efficient detection of intrusions in high-speed network traffic.

It should be strongly noted that the foregoing citations do not in any way state, suggest, or otherwise imply that such claim language is limited in scope to the specific embodiments depicted in the drawings or described in the specification in the instant application.

The Examiner has also argued that such aforementioned claim language would have been obvious and well known to a person skilled in the art, and noted Porras in such regard. Specifically, the Examiner has argued that the motivation to use APIs in order to build the intrusion detection system would have been to take advantage of an already prepared and well tested element to perform part of the required functionality. Appellant respectfully asserts that the alleged obviousness of utilizing APIs does not make appellant's specific claim language obvious, since appellant does not merely claim using APIs, but instead specifically claims "allowing the intrusion detection device to call an application program interface configured to open a protocol decoding application associated with the separate data monitoring device, and...allowing the intrusion detection device to call an application program

interface configured to open an alarm generation application associated with the separate data monitoring device." Thus, each device, as claimed by appellant, is allowed to call API's with specific separate functionality such that the intrusion detection device is allowed to leverage the separate data monitoring device. These claimed features are simply non-existent in both Vaidya and Porras.

In the Examiner's Answer mailed 06/23/2006, the Examiner argued that "applicant's specifications or claims (original or amended) do not point out any advantage in separation of the intrusion detection device and data monitoring device that is distinct from teaching of Vaidya, except for the use of APIs to invoke certain functionalities" and "[t]herefore, examiner did notice the use of API's to call functions in separate modules." Appellant respectfully disagrees with the Examiner's arguments.

Appellant respectfully asserts that Vaidya discloses that "[i]f the virtual processor 36 determines that a network intrusion has occurred, it alerts a reaction module 38, which initiates one of several reactions depending on the nature of the attack" (Col. 6, lines 18-21 – emphasis added). Clearly, the mere disclosure that the virtual processor 36 alerts a reaction module 38 simply fails to even suggest a technique "wherein the application program interfaces allow the intrusion detection device to leverage the separate data monitoring device, ... by allowing the intrusion detection device to call an application program interface configured to open an alarm generation **application** associated with the separate data monitoring device" (emphasis added), as claimed by appellant.

Further, Vaidya discloses that "[t]he virtual processor 36 obtains a data packet from a queue and extracts MAC header information, IP header information, transport header information, and application information from the data packet" (Col. 7, lines 18-21 – emphasis added). However, the mere disclosure by Vaidya that the virtual processor directly obtains information from a data packet in the queue fails to even suggest a technique "wherein the application program interfaces allow the intrusion detection device to leverage the separate data monitoring device, by allowing the intrusion detection device to call an application program interface configured to open a protocol decoding application associated with the separate data monitoring device..." (emphasis added), as claimed by appellant. Clearly, since the virtual processor extracts

information from the data packet in the queue, Vaidya actually *teaches away* from the need for the intrusion detection device to "open a protocol decoding application associated with the separate data monitoring device..." (emphasis added), as claimed by appellant.

Further, the Examiner argued that "using APIs as a means to transfer information between objects or elements of a distributed system is obvious and well known to a person skilled in the art and would have been obviously to a person skilled in the art to use the APIs in order to build the intrusion detection system invented by Vaidya." In addition, the Examiner argued that "[t]herefore, Examiner has established that Vaidya's anticipation of application programming interfaces as one of inherency, because Vaidya's system does have separate modules that exchange information among one another."

Appellant respectfully disagrees with the Examiner's arguments and asserts that Vaidya merely discloses a "data collector 10 [which] includes a communication module 34," "a configuration builder module 32," "a stateful dynamic signature inspection (SDSI) virtual processor 36," and a "reaction module 38" (Col. 6, lines 1-26). In addition, Vaidya discloses the use of caches to facilitate transferring data in between modules as seen in items 32, 42, and 36 of Figure 4. For example, Vaidya discloses that "[t]he configuration builder module 32 temporarily stores the applicable attack signature profiles in an instruction cache 42" and that "[t]he virtual processor 36 processes the attack signature profiles to determine whether the packet is associated with a network intrusion attempt" (Col. 7, lines 32-36 – emphasis added).

Appellant asserts that the fact that a certain result or characteristic may occur or be present in the prior art is not sufficient to establish the inherency of that result or characteristic. *In re Rijckaert*, 9 F.3d 1531, 1534, 28 USPQ2d 1955, 1957 (Fed. Cir. 1993); *In re Oelrich*, 666 F.2d 578, 581-82, 212 USPQ 323, 326 (CCPA 1981). "To establish inherency, the extrinsic evidence 'must make clear that the missing descriptive matter is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill. Inherency, however, may not be established by probabilities or possibilities. The mere fact that a certain

thing may result from a given set of circumstances is not sufficient.'" *In re Robertson*, 169 F.3d 743, 745, 49 USPQ2d 1949, 1950-51 (Fed. Cir. 1999).

In view of the limited Vaidya disclosure and the failings thereof highlighted above, there is absolutely no evidence in such reference that makes it clear that such missing descriptive matter is necessarily present in the Vaidya system. In view of the arguments made hereinabove, any inherency argument has been adequately rebutted, and a notice of allowance or a specific prior art showing of such claim features, in combination with the remaining claim elements is respectfully requested. (See MPEP 2112)

Additionally, the Examiner argued that Porras "clearly suggests the use of APIs (paragraphs 40 to 42 and claim 2) in development of intrusion detection systems." Appellant respectfully disagrees with the Examiner's argument and asserts that the paragraphs from Porras relied upon by the Examiner merely disclose that "using the Apache API 42 the application-integrated intrusion detection process 34 integrates intrusion detection at the application layer, e.g., with the web server process 32" and that "[w]hen one writes a module for the Apache web server, one defines global API structures in source code that establish a connection to the server code **during linking**, thus integrating the module with the web server process" (emphasis added). Clearly, the disclosure by Porras that an API is used to integrate intrusion detection at the application layer and that the API allows the intrusion detection module to be linked in with the server code thereby integrating the intrusion detection with the web server process *teaches away* from a technique "wherein the application program interfaces allow the intrusion detection device to leverage the **separate** data monitoring device, by allowing the intrusion detection device to call an application program interface configured to open a protocol decoding application associated with the separate data monitoring device, and by allowing the intrusion detection device to call an application program interface configured to open an alarm generation application associated with the separate data monitoring device" (emphasis added), as claimed by appellant.

With respect to the 102 rejection, the Examiner is reminded that a claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described in a single prior art reference. *Verdegaal Bros. v. Union Oil Co. Of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). Moreover, the identical invention must be shown in as complete detail as contained in the claim. *Richardson v. Suzuki Motor Co.*868 F.2d 1226, 1236, 9USPQ2d 1913, 1920 (Fed. Cir. 1989). The elements must be arranged as required by the claim, as noted above.

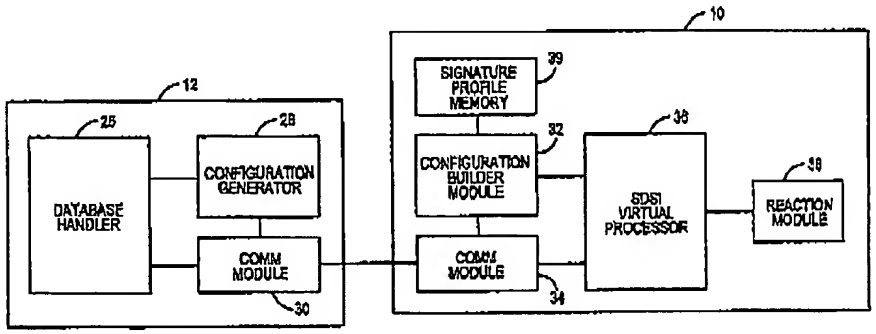
With respect to the 103 rejection, to establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on appellant's disclosure. *In re Vaeck,*947 F.2d 488, 20 USPQ2d 1438 (Fed.Cir.1991). Appellant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, as noted above.

Thus, all of the independent claims are deemed allowable. Moreover, the remaining dependent claims are further deemed allowable, in view of their dependence on such independent claims.

Group #2: Claim 11

With respect to independent Claim 11, appellant incorporates the arguments made hereinabove regarding Group #1. Further, it is noted that Claim 11 includes additional language requiring a technique of "... allowing the intrusion detection device to call at least one application program interface configured to open applications of the data monitoring device; and performing intrusion detection at the intrusion detection device utilizing at least one of the applications of the data

monitoring device" (emphasis added). The Examiner relied upon items 26, 32, 34, 36, and 39 of Fig. 2, and Col. 6, lines 1-11 (see below) from Vaidya to make a prior art showing of such claim language.



(Vaidya, Fig. 2)

"Each data collector 10 includes a communication module 34 for transmitting and receiving information to and from the data repository 12. A configuration builder module 32 assigns a set of signature profiles to each network object and stores data representative of associations between network objects and attack signature profile sets in a signature profile memory 39. The configuration builder module 32 accesses the appropriate attack signature profile sets during operation of the data collector 10 and provides the attack signature profiles to a stateful dynamic signature inspection (SDSI) virtual processor 36. The attack signature profiles include a set of instructions which the virtual processor 36 executes to determine whether a particular data packet is associated with a network intrusion. Although a preferred embodiment of the processor employs the software based virtual processor 36 to execute attack signature profiles, a hardware based processor can be employed in the place of the virtual processor 36." (Vaidya, Col. 6, lines 1-11 - emphasis added)

The Examiner has further argued that "the configuration builder module (item 32) allows the intrusion detection device (item 36) access attack signature profiles stored in signature profile memory (item 39)." However, the excerpts merely suggest a technique where "[t]he configuration builder module 32 accesses the appropriate attack signature profile sets during operation of the data collector 10 and provides the attack signature profiles to a stateful dynamic signature inspection (SDSI) virtual processor 36" (emphasis added). Further, the Examiner has argued that "the communication module (item 34) allows intrusion detection device access the data in database handler (item 26)." Again, appellant respectfully asserts that the excerpt

simply teaches a technique where"[e]ach data collector 10 includes a communication module 34 for transmitting and receiving information to and from the data repository 12" (emphasis added).

To this end, there is clearly not even a suggestion in the above excerpts of a technique of "allowing the intrusion detection device to call at least one application program interface configured to open applications of the data monitoring device; and performing intrusion detection at the intrusion detection device utilizing at least one of the applications of the data monitoring device" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/23/2006, the Examiner argued that "column 6 lines 1-11 clearly show the exchange of data between different modules of Vaidya's system." In addition, the Examiner argues that "this inherently requires the modules to have the capability to initiate an action on a separate device, which is the essence of Application Programming Interfaces." Again, appellant respectfully disagrees with the Examiner's inherency arguments and incorporates the arguments made hereinabove regarding Issue #3, Group #1.

In addition, appellant respectfully asserts that Col. 6, lines 1-11 in Vaidya relied upon by the Examiner merely disclose that "[t]he configuration builder module 32 accesses the appropriate attack signature profile sets during operation of the data collector 10 and provides the attack signature profiles to a stateful dynamic signature inspection (SDSI) virtual processor 36." However, in Figure 4 and in Col. 7, lines 32-36, Vaidya further discloses that "[t]he configuration builder module 32 temporarily stores the applicable attack signature profiles in an instruction cache 42" and that "[t]he virtual processor 36 processes the attack signature profiles to determine whether the packet is associated with a network intrusion attempt" (emphasis added). However, Vaidya's mere disclosure that the configuration builder module 32 stores attack signature profiles in instruction cache 42 for the virtual processor 36 fails to even suggest "allowing the intrusion detection device to call at least one application program interface configured to open applications of the data monitoring device; and performing intrusion detection at the intrusion

detection device utilizing at least one of the applications of the data monitoring device" (emphasis added), as claimed by appellant.

Again, in view of the limited Vaidya disclosure and the failings thereof highlighted above, there is absolutely no evidence in such reference that makes it clear that the admitted missing descriptive matter is necessarily present in the Vaidya system. In view of the arguments made hereinabove, any inherency argument has been adequately rebutted, and a notice of allowance or a specific prior art showing of such claim features, in combination with the remaining claim elements is respectfully requested. (See MPEP 2112)

Group #3: Claim 16

With respect to dependent Claim 16, the Examiner has relied on the following excerpts from the Vaidya reference to make a prior art showing of appellant's claimed technique "wherein the application program interfaces provide parsing of signatures used in signature matching" (see Claim 16).

> "The sequential signature attribute refers to multiple expressions which are sequentially executed on successively transmitted data packets associated with an application session. If each of the expressions detects the event it was designed to detect, a network intrusion has been detected.
>
> A more formal description of an attack signature in a loose BNR parsing grammar follows:
>
> Pattern := Hex or ASCII string of characters
> Offset := integer
> Protocol := one of the communication protocols, ie. MAC-layer
> Network-layer, Transport-layer, or Application-layer
> Extract_Type:= Byte, Word, Long Word or String
> Header_Field:= Predefined keywords for communication protocol header fields
> Variable_Name:= ASCII character string Name
> SP :=<Pattern, Offset, Protocol> . . . Search Primitive
> VP :=<Extract_Type,. Offset, Protocol> . . . Value Primitive
> OP :=<Logical> .|. <Arithmetic> .|. <Bit-wide> | <Association> | … Operators
> Basic_Expression:= <SP>.|.<OP>.|.<Header_Field.|.<SP OP SP>.

```
                |. <SP OP VP .|. <SP OP Header_Field>
      Assignment := <Variable_Name> "=" <Basic_Expression>
      Complex_Expression := {(<Basic_Expression> OP <Basic
                Expression>) . . . }
      Expression                    :=     <Complex_Expression>     .|.
<Complex_Expression>";"
                {(<Assignment>";") . . . }
      Signature_Attributes := <Simple> .|. <Counter-Timer-Based>.|.
                <Sequential-occurrence>
      Attack_Signature := <Signature_Attribute> { <Expression> . . .
      .}"

      (Vaidya, Col. 10, lines 17-45 - emphasis added)
```

Appellant respectfully asserts that the excerpt above simply does not meet all of appellant's claim limitations. Specifically, the Vaidya excerpt teaches "...attack signature in a loose BNR parsing grammar..." and that the "...sequential signature attribute refers to multiple expressions which are sequentially executed on successively transmitted data packets..." The BNR parsing grammar and multiple expressions, as described by Vaidya, clearly do not, however, meet appellant's specific claim language, since Vaidya fails to even suggest a technique "wherein the application program interfaces provide parsing of signatures used in signature matching" (emphasis added), as claimed by appellant.

In the Examiner's Answer mailed 06/23/2006, the Examiner argued that "the mention[ed] excerpt of Vaidya (Column 10, lines 17-45) clearly shows an attack signature parsed" and "[t]herefore, parsing attack signatures are clearly disclosed by Vaidya." Appellant respectfully disagrees and asserts that the excerpt from Vaidya merely discloses a "formal description of an attack signature in a loose BNR parsing grammar." In addition, Col. 10, lines 46-48 in Vaidya disclose "a method for processing attack signature profiles includes obtaining an attack signature profile from the instruction cache 42 in step 122" (emphasis added). However, merely disclosing the description of an attack signature in loose BNR parsing grammar and that the attack signature profile is obtained from the instruction cache 42 fails to even suggest a technique "wherein the application program interfaces provide parsing of signatures used in signature matching" (emphasis added), as claimed by appellant. Clearly, since the attack signature is in the instruction cache 42, there is not even a need to use "application program interfaces [to] provide parsing of signatures used in signature

matching" (emphasis added), as claimed by appellant. Thus, Vaidya actually *teaches away* in this regard.

In the event a telephone conversation would expedite the prosecution of this application, the Examiner may reach the undersigned at (408) 971-2573. For payment of any additional fees due in connection with the filing of this paper, the Commissioner is authorized to charge such fees to Deposit Account No. 50-1351 (Order No. NAI1P317/01.185.01).

Respectfully submitted,

By: _____ Date: 8/23/06

Kevin J. Zilka

Reg. No. 41,429

Zilka-Kotab, P.C.
P.O. Box 721120
San Jose, California 95172-1120
Telephone: (408) 971-2573
Facsimile: (408) 971-4660